CLAIMS

1. One or more computer readable media having stored thereon a program that, when executed by one or more processors, causes the one or more processors to perform acts including:

identifying a plurality of key instructions in a function;

inserting into the function, for each of the plurality of key instructions, an extra instruction that modifies a register based at least in part on the corresponding key instruction;

identifying a set of inputs to the function; and

determining a checksum for the function based at least in part on mapping contents of the register to the set of inputs.

2. One or more computer readable media as recited in claim 1, wherein the identifying a plurality of key instructions comprises identifying, as a key instruction, each instruction in the function that possibly modifies a register or a flag.

3. One or more computer readable media as recited in claim 1, wherein the identifying a plurality of key instructions comprises identifying, as the plurality of key instructions, a plurality of instructions that each modify one or more registers or one or more flags.

4. One or more computer readable media as recited in claim 1, wherein the inserting comprises inserting each extra instruction in a location within the function so that the extra instruction is executed if the corresponding key instruction is executed.

5. One or more computer readable media as recited in claim 1, wherein the inserting comprises inserting each extra instruction in a location within the function so that the extra instruction is executed after the corresponding key instruction is executed.

6. One or more computer readable media as recited in claim 1, wherein the inserting comprises inserting the extra instructions into the function without altering the control flow of the function.

7. One or more computer readable media as recited in claim 1, wherein the inserting comprises inserting, for at least one of the plurality of key instructions, a plurality of extra instructions that modify one or more registers.

8. One or more computer readable media as recited in claim 1, wherein the identifying a set of inputs comprises identifying a set of input patterns to the function that result in different valid computation paths in the function being taken.

9. One or more computer readable media as recited in claim 1, wherein the identifying a set of inputs comprises identifying a set of input patterns to the function that result in all valid computation paths in the function being taken.

10. One or more computer readable media as recited in claim 1, wherein the determining comprises determining as the checksum both an initial value (x_0) and a calculated value (*Cks*), wherein the initial value is a first input of the set of inputs, and wherein the calculated value is calculated according to the following process:

```
Start with $x = x_0$
Cks := $f(x_0)$ XOR $x_0$
For i=1 to K do
     $x_i$  := $g(f(x_i-1))$
     Cks += $f(x_i)$ XOR $x_i$
End for
```

wherein *K* is the number of inputs in the set of inputs and *g* represents the mapping function.

11. One or more computer readable media as recited in claim 1, wherein the function is part of a software program.

12. A method comprising:
generating a checksum on bytes of a digital good without reading the bytes.

13. A method as recited in claim 12, wherein the generating comprises:
identifying a plurality of key instructions in a function;

inserting into the function, for each of the plurality of key instructions, an extra instruction that modifies a register based at least in part on the corresponding key instruction;

identifying a set of inputs to the function; and

determining a checksum for the function by mapping contents of the register to the set of inputs.

14. A method as recited in claim 13, wherein the identifying comprises identifying, as a key instruction, each instruction in the function that possibly modifies a register or a flag.

15. One or more computer-readable memories comprising computer-readable instructions that, when executed by a processor, direct a computer system to perform the method as recited in claim 13.

16. A method comprising:

inserting, into a segment of a digital good, a plurality of instructions that modify a register;

identifying a set of inputs to the segment; and

determining a checksum value for the segment based at least in part on both the set of inputs and the register contents.

17. A method as recited in claim 16, wherein the inserting comprises:

identifying a plurality of key instructions in the segment; and inserting into the segment, for each of the plurality of key instructions, an extra instruction that modifies a register based at least in part on the corresponding key instruction.

18. A method as recited in claim 17, wherein the identifying a comprises identifying, as a key instruction, each instruction in the segment that possibly modifies a register or a flag.

19. A method as recited in claim 17, wherein the identifying a plurality of key instructions comprises identifying, as the plurality of key instructions, a plurality of instructions that each modify one or more registers or one or more flags.

20. A method as recited in claim 16, wherein the inserting comprises inserting each of the plurality of instructions in a location within the segment so that the instruction is executed if a corresponding key instruction is executed.

21. A method as recited in claim 16, wherein the inserting comprises inserting each of the plurality of instructions in a location within the segment so that the instruction is executed after a corresponding key instruction is executed.

22. A method as recited in claim 16, wherein the inserting comprises inserting the extra instructions into the segment without altering a control flow of the segment.

23. A method as recited in claim 16, wherein the inserting comprises inserting a plurality of extra instructions that modify one or more registers.

24. A method as recited in claim 16, wherein the identifying a set of inputs comprises identifying a set of input patterns to the segment that result in different valid computation paths in the segment being taken.

25. A method as recited in claim 16, wherein the identifying a set of inputs comprises identifying a set of input patterns to the segment that result in all valid computation paths in the segment being taken.

26. A method as recited in claim 16, wherein the determining comprises determining as the checksum value both an initial value (x_0) and a calculated value (*Cks*), wherein the initial value is a first input of the set of inputs, and wherein the calculated value is calculated according to the following process:

```
Start with x = $x_0$
Cks := f($x_0$) XOR $x_0$
For i=1 to K do
    $x_i$  := g(f($x_i$-1))
    Cks += f($x_i$) XOR $x_i$
End for
```

wherein *K* is the number of inputs in the set of inputs and *g* represents the mapping function.

27. A method as recited in claim 16, wherein the digital good comprises a software program.

28. One or more computer-readable memories comprising computer-readable instructions that, when executed by a processor, direct a computer system to perform the method as recited in claim 16.

/

29. A production system, comprising:

a memory to store an original program; and

a production server equipped with an oblivious checking protection tool that is used to augment the original program for protection purposes, the production server being configured to identify a plurality of segments in the original program and apply oblivious checking to each of the plurality of segments.

30. A production system as recited in claim 29, wherein the production server is to apply oblivious checking to each of the plurality of segments by:

inserting, into the segment, a plurality of instructions that modify a register;

identifying a set of inputs to the segment; and

determining a checksum value for the segment based at least in part on both the set of inputs and the register content that results from applying the set of inputs to the segment.

31. A production system as recited in claim 29, wherein the inserting comprises:

identifying a plurality of key instructions in the segment; and

inserting into the segment, for each of the plurality of key instructions, an extra instruction that modifies a register based at least in part on the corresponding key instruction.

32. A production system as recited in claim 31, wherein the identifying a comprises identifying, as a key instruction, each instruction in the segment that possibly modifies a register or a flag.

33. A production system as recited in claim 29, wherein the inserting comprises inserting the extra instructions into the segment without altering a control flow of the segment.

34. A client-server system, comprising:

a production server to apply oblivious checking to a program to produce a protected program; and

a client to store and execute the protected program, the client being configured to evaluate the protected program to determine whether the protected program has been tampered with.

35. A client-server system as recited in claim 34, wherein the production server is to apply oblivious checking to the program by:

inserting, into a segment of the program, a plurality of instructions that modify a register;

identifying a set of inputs to the segment; and

determining a checksum value for the segment based at least in part on both the set of inputs and the register content that results from applying the set of inputs to the segment.

36. A client-server system as recited in claim 34, wherein the client is to evaluate the protected program by:

generating a checksum value for each of a plurality of segments of the program based at least in part on both a set of inputs to the segment and the content of a register that results from applying the set of inputs to the segment;

comparing, for each of the plurality of segments, the generated checksum value to a stored checksum value corresponding to the segment; and

determining the protected program has been tampered with if the generated checksum value for any one or more of the plurality of segments does not match the stored checksum value for the segment.

37. A client-server system as recited in claim 34, wherein the program comprises a software program.

38. One or more computer readable media having stored thereon a plurality of instructions that, when executed by one or more processors, causes the one or more processors to perform acts including:

generating a checksum value for a segment of a digital good based at least in part on both a set of inputs to the segment and the content of a register that results from applying the set of inputs to the segment;

comparing the generated checksum value to a stored checksum value corresponding to the segment; and

determining that the digital good has been tampered with if the generated checksum value does not match the stored checksum value.

39. One or more computer readable media as recited in claim 38, wherein the plurality of instructions further cause the one or more processors to perform acts including repeating the generating, comparing, and determining for a plurality of segments of the digital good.

40. One or more computer readable media as recited in claim 38, wherein the digital good comprises a software program.